Chapter 7
Reed–Solomon Codes and Binary Transmission

7.1 Introduction

Reed–Solomon codes named after Reed and Solomon [9] following their publication in 1960 have been used together with hard decision decoding in a wide range of applications. Reed–Solomon codes are maximum distance separable (MDS) codes and have the highest possible minimum Hamming distance. The codes have symbols from $\mathbb{F}_q$ with parameters $(q-1,\ k,\ q-k)$. They are not binary codes but frequently are used with $q = 2^m$, and so there is a mapping of residue classes of a primitive polynomial with binary coefficients [6] and each element of $\mathbb{F}_{2^m}$ is represented as a binary m-tuple. Thus, binary codes with code parameters $(m[2^m-1],\ km,\ 2^m-k)$ can be constructed from Reed–Solomon codes. Reed–Solomon codes can be extended in length by up to two symbols and in special cases extended in length by up to three symbols. In terms of applications, they are probably the most popular family of codes.

Researchers over the years have tried to come up with an efficient soft decision decoding algorithm and a breakthrough in hard decision decoding in 1997 by Madhu Sudan [10], enabled more than $\frac{2^m-k}{2}$ errors to be corrected with polynomial time complexity. The algorithm was limited to low rate Reed–Solomon codes. An improved algorithm for all code rates was discovered by Gursuswami and Sudan [3] and led to the Guruswami and Sudan algorithm being applied in a soft decision decoder by Kötter and Vardy [5]. A very readable, tutorial style explanation of the Guruswami and Sudan algorithm is presented by McEliece [7]. Many papers followed, discussing soft decision decoding of Reed–Solomon codes [1] mostly featuring simulation results of short codes such as the (15, 11, 5) and the (31, 25, 7) code. Binary transmission using baseband bipolar signalling or binary phase shift keying (BPSK) [8] and the additive white gaussian noise (AWGN) channel is most common. Some authors have used quadrature amplitude modulation (QAM) [8] with 2^m levels to map to each $\mathbb{F}_{2^m}$ symbol [5]. In either case, there is a poor match between

M. Tomlinson et al., *Error-Correction Coding and Decoding*,
Signals and Communication Technology,
DOI 10.1007/978-3-319-51103-0_7

the modulation method and the error-correcting code. The performance achieved is not competitive compared to other error-correcting code arrangements. For binary transmission, a binary error-correcting code should be used and not a symbol-based error-correcting code. For QAM and other multilevel signalling, better performance is obtained by applying low-rate codes to the least significant bits of received symbols and high-rate codes to the most significant bits of received symbols. Applying a fixed-rate error-correcting code to all symbol bits is the reason for the inefficiency in using Reed–Solomon codes on binary channels.

Still, these modulation methods do provide a means of comparing different decoder arrangements for RS codes. This theme is explored later in Sect. 7.3 where soft decision decoding of RS codes is explored.

7.2 Reed–Solomon Codes Used with Binary Transmission-Hard Decisions

Whilst RS codes are very efficient codes, being MDS codes, they are not particularly well suited to the binary channel as it will become apparent from the results presented below. Defining the RS code over $\mathbb{F}_{2^m}$, RS codes extended with a single symbol are considered with length $n = 2^m$, with k information symbols, and with $d_{min} = n-k+1$. The length in bits, $n_b = mn$ and there are k_b information bits with $k_b = km$.

The probability of a symbol error with binary transmission and the AWGN channel is

$$p_s = 1 - \left(1 - \frac{1}{2} erfc\left(\sqrt{\frac{k}{n}\frac{E_b}{N_0}}\right)\right)^m \tag{7.1}$$

The RS code can correct t errors where $t = \left\lfloor \frac{n-k+1}{2} \right\rfloor$. Accordingly, a decoder error occurs if there are more than t symbol errors and the probability of decoder error, p_C is given by

$$p_C = \sum_{i=t+1}^{n} \frac{n!}{(n-i)!i!} p_s^i (1-p_s)^{n-i} \tag{7.2}$$

As a practical example, we will consider the (256, 234, 23) extended RS code. Representing each $\mathbb{F}_{2^8}$ symbol as a binary 8 tuple the RS code becomes a (2048, 1872, 23) binary code. The performance with hard decisions is shown in Fig. 7.1 as a function of $\frac{E_b}{N_0}$. This code may be directly compared to the binary (2048, 1872, 33) Goppa code since their lengths and code rates are identical. The decoder error probability for the binary Goppa code is given by

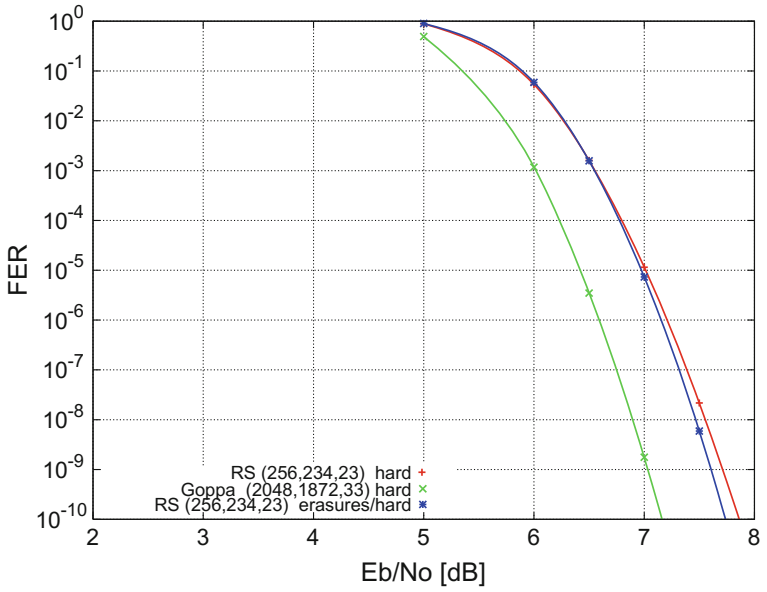

Fig. 7.1 Comparison of hard decision decoding of the (256, 234, 23) RS code compared to the (2048, 1872, 33) Goppa code (same code length in bits and code rate)

$$p_C = \sum_{i=t_G+1}^{nm} \frac{(nm)!}{(nm-i)!i!} \left(\frac{1}{2} erfc \sqrt{\frac{k}{n}\frac{E_b}{N_0}}\right)^i \left(1 - \frac{1}{2} erfc \sqrt{\frac{k}{n}\frac{E_b}{N_0}}\right)^{nm-i} \tag{7.3}$$

where $t_G = \left\lfloor \frac{d_{min}+1}{2} \right\rfloor$ for the binary Goppa code.

The comparison in performance is shown in Fig. 7.1 and it can be seen that the Goppa code is approximately 0.75dB better than the RS code at 1×10^{-10} frame error rate.

It is interesting to speculate whether the performance of the RS code could be improved by using 3-level quantisation of the channel bits and erasing symbols if any of the bits within a symbol are erased. The probabilities of a bit erasure p_{erase} and bit error p_b for 3-level quantisation are given in Chap. 3, Eqs. (3.41) and (3.42) respectively, but note that a lower threshold needs to be used for best performance with these code parameters, $\sqrt{E_s} - 0.2 \times \sigma$ instead of $\sqrt{E_s} - 0.65 \times \sigma$. The probability of a symbol erasure, $p_{S\,erase}$ is given by

$$p_{S\,erase} = 1 - (1 - p_{erase})^m \tag{7.4}$$

and the probability of a symbol error, $p_{S\,error}$ is given by

$$p_{S\,error} = 1 - \left(1 - (1 - p_{erase})^m\right) - (1 - p_b)^m \tag{7.5}$$

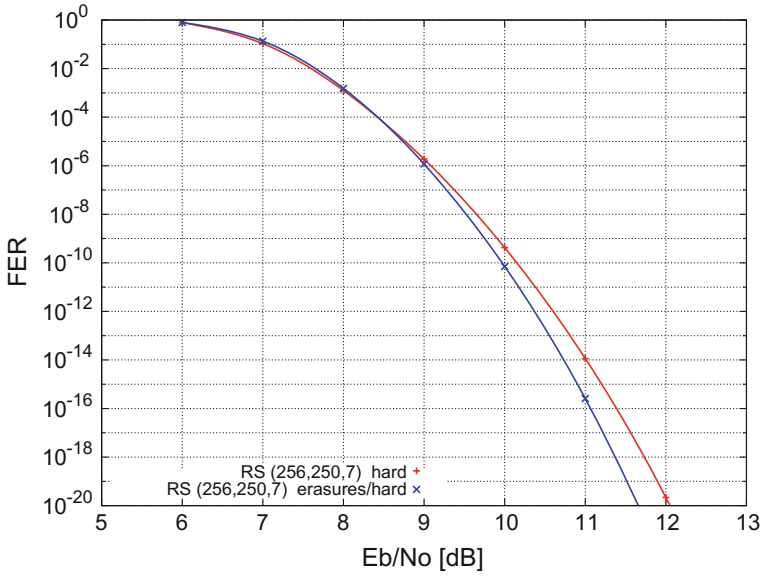

Fig. 7.2 Comparison of hard decision and erasure decoding of the (256, 250, 7) RS code for the binary channel

and

$$p_{S\,error} = (1 - p_{erase})^m - (1 - p_b)^m \tag{7.6}$$

For each received vector, provided the number of errors t and the number of erasures s such that $2t + s \leq n - k$, then the received vector will be decoded correctly. A decoder error occurs if $2t + s > n - k$.

The probability distribution of errors and erasures in the received vector, $e(z)$ may be easily found by defining a polynomial $p(z)$ and raising it to the power of n, the number of symbols in a codeword.

$$e(z) = \left(1 - p_{S\,error} - p_{S\,erase} + p_{S\,erase} z^{-1} + p_{S\,error} z^{-2}\right)^n \tag{7.7}$$

The probability of decoder error p_C is simply found from $e(z)$ by summing all coefficients of z^{-i} where i is greater than $n - k$. This is very straightforward with a symbolic mathematics program such as Mathematica. The results for the RS (256, 234, 23) code are shown in Fig. 7.1. It can be seen that there is an improvement over the hard decision case but it is rather marginal.

A rather more convincing case is shown in Fig. 7.2 for the RS (256, 250, 7) code where the performance is shown down to frame error rates of 1×10^{-20}. In this case, there is an improvement of approximately 0.4 dB.

It has already been established that for the binary transmission channel, the RS codes based on $GF(2^m)$, do not perform as well as a binary designed code with the same code parameters. The problem is that bit errors occur independently and it only takes a single bit error to cause a symbol error. Thus, the code structure, being symbol based, is not well matched to the transmission channel. Another way of looking at this is to consider the Hamming distance. For the binary (2048, 1872) codes considered previously, the RS-based code turns out to have a binary Hamming distance of 23 whilst the binary Goppa code has a Hamming distance of 33. However, there is a simple method of modifying RS codes to produce good binary codes as discussed in Chap. 6. It is a code concatenation method best suited for producing symbol-based binary codes whereby a single overall binary parity check is added to each binary m-tuple representing each symbol. Starting with a RS $(n, k, n-k-1)$ code, adding the overall binary parity checks produces a $(n[m+1], km, 2[n-k-1])$ binary code. Now the minimum weight of each symbol is 2, producing a binary code with twice the minimum Hamming distance of the original RS code. Kasahara [4] realised that in some cases an additional information bit may be added by adding the all $1's$ codeword to the generator matrix. Some best known codes are constructed in this way as discussed in Chap. 6. One example is the (161, 81, 23) binary code [6].

7.3 Reed–Solomon Codes and Binary Transmission Using Soft Decisions

RS codes applied to the binary transmission channel will now be considered using unquantised soft decision decoding. The best decoder to use is the modified Dorsch decoder, discussed in Chap. 15, because it provides near maximum likelihood decoding. However when used with codes having a significant coding gain, the code length needs to be typically less than 200 bits.

We will consider augmented, extended RS codes constructed from $GF(2^m)$. The length is $2^m + 1$ and these are Maximum Distance Separable (MDS) codes with parameters $(2^m + 1, k, 2^{m+1} - k)$. Moreover, the general case is that augmented, extended RS codes may be constructed using any Galois Field $GF(q)$ with parameters $(q+1, k, q+2-k)$ [6]. Denoting the q field elements as $0,\ \alpha_0,\ \alpha_1,\ \alpha_2, \ldots\ \alpha_{q-2}$, the parity-check matrix is given by

$$\mathbf{H} = \begin{bmatrix} \alpha_0^j & \alpha_1^j & \alpha_2^j & \ldots & \alpha_{q-2}^j & 1 & 0 \\ \alpha_0^{j+1} & \alpha_1^{j+1} & \alpha_2^{j+1} & \ldots & \alpha_{q-2}^{j+1} & 0 & 0 \\ \alpha_0^{j+2} & \alpha_1^{j+2} & \alpha_2^{j+2} & \ldots & \alpha_{q-2}^{j+2} & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \alpha_0^{j+q-k-1} & \alpha_1^{j+q-k-1} & \alpha_2^{j+q-k-1} & \ldots & \alpha_{q-2}^{j+q-k-1} & 0 & 0 \\ \alpha_0^{j+q-k} & \alpha_1^{j+q-k} & \alpha_2^{j+q-k} & \ldots & \alpha_{q-2}^{j+q-k} & 0 & 1 \end{bmatrix}$$

Table 7.1 $GF(32)$ non-zero extension field elements defined by $1 + \alpha^2 + \alpha^5 = 0$

$\alpha^0 = 1$	$\alpha^{16} = 1 + \alpha + \alpha^3 + \alpha^4$
$\alpha^1 = \alpha$	$\alpha^{17} = 1 + \alpha + \alpha^4$
$\alpha^2 = \alpha^2$	$\alpha^{18} = 1 + \alpha$
$\alpha^3 = \alpha^3$	$\alpha^{19} = \alpha + \alpha^2$
$\alpha^4 = \alpha^4$	$\alpha^{20} = \alpha^2 + \alpha^3$
$\alpha^5 = 1 + \alpha^2$	$\alpha^{21} = \alpha^3 + \alpha^4$
$\alpha^6 = \alpha + \alpha^3$	$\alpha^{22} = 1 + \alpha^2 + \alpha^4$
$\alpha^7 = \alpha^2 + \alpha^4$	$\alpha^{23} = 1 + \alpha + \alpha^2 + \alpha^3$
$\alpha^8 = 1 + \alpha^2 + \alpha^3$	$\alpha^{24} = \alpha + \alpha^2 + \alpha^3 + \alpha^4$
$\alpha^9 = \alpha + \alpha^3 + \alpha^4$	$\alpha^{25} = 1 + \alpha^3 + \alpha^4$
$\alpha^{10} = 1 + \alpha^4$	$\alpha^{26} = 1 + \alpha + \alpha^2 + \alpha^4$
$\alpha^{11} = 1 + \alpha + \alpha^2$	$\alpha^{27} = 1 + \alpha + \alpha^3$
$\alpha^{12} = \alpha + \alpha^2 + \alpha^3$	$\alpha^{28} = \alpha + \alpha^2 + \alpha^4$
$\alpha^{13} = \alpha^2 + \alpha^3 + \alpha^4$	$\alpha^{29} = 1 + \alpha^3$
$\alpha^{14} = 1 + \alpha^2 + \alpha^3 + \alpha^4$	$\alpha^{30} = \alpha + \alpha^4$
$\alpha^{15} = 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4$	

There are $q - k + 1$ rows of the matrix corresponding to the $q - k + 1$ parity symbols of the code. Any of the $q - k + 1$ columns form a Vandermonde matrix and the matrix is non-singular which means that any set of $q - k + 1$ symbols of a codeword may be erased and solved using the parity-check equations. Thus, the code is MDS. The columns of the parity-check matrix may be permuted into any order and any set of s symbols of a codeword may be defined as parity symbols and permanently erased. Thus, their respective columns of $\mathbf{H}$ may be deleted to form a shortened $(2^m + 1 - s, k, 2^{m+1} - s - k)$ MDS code. This is an important property of MDS codes, particularly for their practical realisation in the form of augmented, extended RS codes because it enables efficient implementation in applications such as incremental redundancy systems, discussed in Chap. 17, and network coding. Using the first $q - 1$ columns of $\mathbf{H}$, and setting $\alpha_0,\ \alpha_1,\ \alpha_2, \ldots\ \alpha_{q-2}$ equal to $\alpha^0,\ \alpha^1,\ \alpha^2, \ldots\ \alpha^{q-2}$, where α is a primitive element of $GF(q)$ a cyclic code may be constructed, which has advantages for encoding and decoding implementation.

We will consider the shortened RS code (30, 15, 16) constructed from the $GF(2^5)$ extension field with $\mathbf{H}$ constructed using $j = 0$ and α being the primitive root of $1 + x^2 + x^5$. The $GF(32)$ extension field table is given in Table 7.1 based on the primitive polynomial $1 + x^2 + x^5$ so that $1 + \alpha^2 + \alpha^5 = 0$, modulo $1 + x^{31}$.

The first step in the construction of the binary code is to construct the parity-check matrix for the shortened RS code (30, 15, 16) which is

$$\mathbf{H}_{(30,15)} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \alpha & \alpha^2 & \dots & \alpha^{29} \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{27} \\ 1 & \alpha^3 & \alpha^6 & \dots & \alpha^{25} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \alpha^{13} & \alpha^{26} & \dots & \alpha^5 \\ 1 & \alpha^{14} & \alpha^{28} & \dots & \alpha^3 \end{bmatrix}$$

Each element of this parity-check matrix is to be replaced with a 5×5 matrix in terms of the base field, which in this case is binary. First, the number of rows are expanded to form $\mathbf{H}_{(30,75)}$ given by matrix (7.8). The next step is to expand the columns in terms of the base field by substituting for powers of α using Table 7.1. For example, if an element of the parity-check matrix $\mathbf{H}_{(30,75)}$ is, say α^{26}, then this is replaced by $1+\alpha+\alpha^2+\alpha^4$ which in binary is 11101. Proceeding in this way the binary matrix $\mathbf{H}_{(150,75)}$ is produced (some entries have been left as they were to show the procedure partly completed) as in matrix (7.9).

$$\mathbf{H}_{(30,75)} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ \alpha & \alpha & \alpha & \dots & \alpha \\ \alpha^2 & \alpha^2 & \alpha^2 & \dots & \alpha^2 \\ \alpha^3 & \alpha^3 & \alpha^3 & \dots & \alpha^3 \\ \alpha^4 & \alpha^4 & \alpha^4 & \dots & \alpha^4 \\ 1 & \alpha & \alpha^2 & \dots & \alpha^{29} \\ \alpha & \alpha^2 & \alpha^3 & \dots & \alpha^{30} \\ \alpha^2 & \alpha^3 & \alpha^4 & \dots & 1 \\ \alpha^3 & \alpha^4 & \alpha^5 & \dots & \alpha \\ \alpha^4 & \alpha^5 & \alpha^6 & \dots & \alpha^2 \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{27} \\ \alpha & \alpha^3 & \alpha^5 & \dots & \alpha^{28} \\ \alpha^2 & \alpha^4 & \alpha^6 & \dots & \alpha^{29} \\ \alpha^3 & \alpha^5 & \alpha^7 & \dots & \alpha^{30} \\ \alpha^4 & \alpha^6 & \alpha^8 & \dots & 1 \\ 1 & \alpha^3 & \alpha^6 & \dots & \alpha^{25} \\ \alpha & \alpha^4 & \alpha^7 & \dots & \alpha^{26} \\ \alpha^2 & \alpha^5 & \alpha^8 & \dots & \alpha^{27} \\ \alpha^3 & \alpha^6 & \alpha^9 & \dots & \alpha^{28} \\ \alpha^4 & \alpha^7 & \alpha^{10} & \dots & \alpha^{27} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \alpha^{14} & \alpha^{28} & \dots & \alpha^3 \\ \alpha & \alpha^{15} & \alpha^{29} & \dots & \alpha^4 \\ \alpha^2 & \alpha^{16} & \alpha^{30} & \dots & \alpha^5 \\ \alpha^3 & \alpha^{17} & 1 & \dots & \alpha^6 \\ \alpha^4 & \alpha^{18} & \alpha & \dots & \alpha^7 \end{bmatrix} \tag{7.8}$$

$$\mathbf{H}_{(150,75)} = \begin{bmatrix}
10000\ 10000\ 10000 \ldots 10000 \\
01000\ 01000\ 01000 \ldots 01000 \\
00100\ 00100\ 00100 \ldots 00100 \\
00010\ 00010\ 00010 \ldots 00010 \\
00001\ 00001\ 00001 \ldots 00001 \\
10000\ 01000\ 00100 \ldots 10010 \\
01000\ 00100\ 00010 \ldots \alpha^{30} \\
00100\ 00010\ 00001 \ldots 10000 \\
00010\ 00001\ 10100 \ldots 01000 \\
00001\ 10100\ 01010 \ldots 00100 \\
10000\ 00100\ 00001 \ldots 11010 \\
01000\ 00010\ 10100 \ldots 01101 \\
00100\ 00001\ 01010 \ldots 10010 \\
00010\ 10100\ 00101 \ldots \alpha^{30} \\
00001\ 01010\ 10110 \ldots 10000 \\
10000\ 00010\ 01010 \ldots 10011 \\
01000\ 00001\ 00101 \ldots 11101 \\
00100\ 10100\ 10110 \ldots 11010 \\
00010\ 01010\ 01011 \ldots 01101 \\
00001\ 00101\ \alpha^{10} \ldots 11010 \\
\vdots\ \ \vdots\ \ \vdots\ \ \vdots\ \ \vdots \\
10000\ \alpha^{14}\ 01101 \ldots 00010 \\
01000\ \alpha^{15}\ 10010 \ldots 00001 \\
00100\ \alpha^{16}\ \alpha^{30} \ldots 10100 \\
00010\ \alpha^{17}\ 1 \ldots 01010 \\
00001\ \alpha^{18}\ \alpha \ldots 00101
\end{bmatrix} \tag{7.9}$$

The resulting binary code is a (150, 75, 16) code with the d_{min} the same as the symbol-based RS (30, 15, 16) code. As observed by MacWilliams [6], changing the basis can increase the d_{min} of the resulting binary code, and making $j = 3$ in the RS parity-check matrix above produces a (150, 75, 19) binary code.

A (150, 75, 22) binary code with increased d_{min} can be constructed using the overall binary parity-check concatenation as discussed above. Starting with the (25, 15, 11) RS code, an overall parity check is added to each symbol, producing a parity-check matrix, $\mathbf{H}_{(150,75,22)}$ given by matrix (7.10). We have constructed two binary (150, 75) codes from RS codes. It is interesting to compare these codes to the known best code of length 150 and rate $\frac{1}{2}$. The known, best codes are to be found in a database [2] and the best (150, 75) code has a d_{min} of 23 and is derived by shortening by one bit (by deleting the x^{150} coordinate from the $\mathbf{G}$ matrix) of the (151, 76, 23) cyclic code whose generator polynomial is

$$\begin{bmatrix}
100001\ 100001\ 100001 \ldots 100001 \\
010001\ 010001\ 010001 \ldots 010001 \\
001001\ 001001\ 001001 \ldots 001001 \\
000101\ 000101\ 000101 \ldots 000101 \\
000011\ 000011\ 000011 \ldots 000011
\end{bmatrix}$$

$$\mathbf{H}_{(150,75,22)} = \begin{bmatrix}
100001\ 010001\ 001001 \ldots 100100 \\
010001\ 001001\ 000101 \ldots 010010 \\
001001\ 000101\ 000011 \ldots 100001 \\
000101\ 000011\ 101000 \ldots 010001 \\
000011\ 101000\ 010100 \ldots 001001 \\
100001\ 001001\ 000011 \ldots 110101 \\
010001\ 000101\ 101000 \ldots 011011 \\
001001\ 000011\ 010100 \ldots 100100 \\
000101\ 101000\ 001010 \ldots 010010 \\
000011\ 010100\ 101101 \ldots 100001 \\
100001\ 000101\ 010100 \ldots 100111 \\
010001\ 000011\ 001010 \ldots 111010 \\
001001\ 101000\ 101101 \ldots 110101 \\
000101\ 010100\ 010111 \ldots 011011 \\
000011\ 001010\ 100010 \ldots 110101 \\
\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\
100001\ 101110\ 011011 \ldots 000101 \\
010001\ 111111\ 100100 \ldots 000011 \\
001001\ 110110\ 010010 \ldots 101000 \\
000101\ 110011\ \ 1 \ \ \ldots 010100 \\
000011\ 110000\ 010001 \ldots 001010
\end{bmatrix} \tag{7.10}$$

$$\begin{aligned} g(x) = {} & 1 + x^3 + x^5 + x^8 + x^{10} + x^{11} + x^{14} + x^{15} + x^{17} + x^{19} + x^{20} + x^{22} \\ & + x^{25} + x^{27} + x^{28} + x^{30} + x^{31} + x^{34} + x^{36} + x^{37} + x^{39} + x^{40} + x^{45} + x^{46} \\ & + x^{48} + x^{50} + x^{52} + x^{59} + x^{60} + x^{63} + x^{67} + x^{70} + x^{73} + x^{74} + x^{75} \end{aligned} \tag{7.11}$$

These three binary codes, the RS-based (150, 75, 19) and (150, 75, 22) codes together with the (150, 75, 23) shortened cyclic code have been simulated using binary transmission for the AWGN channel. The decoder used is a modified Dorsch decoder set to evaluate 2×10^7 codewords per received vector. This is a large number of codewords and is sufficient to ensure that quasi-maximum likelihood performance is obtained. In this way, the true performance of each code is revealed rather than any shortcomings of the decoder.

The results are shown in Fig. 7.3. Also shown in Fig. 7.3, for comparison purposes, is the sphere packing bound and the erasure-based binomial bound discussed in Chap. 1. Interestingly, all three codes have very good performance and are very close to the erasure-based binomial bound. Although not close to the sphere packing bound, this bound is for non-binary codes and there is an asymptotic loss of 0.187 dB for rate $\frac{1}{2}$ binary codes in comparison to the sphere packing bound as the code length extends towards ∞.

Comparing the three codes, no code has the best overall performance over the entire range of $\frac{E_b}{N_0}$, and, surprisingly the d_{min} of the code is no guide. The reason for this can be seen from the Hamming distances of the codewords decoded in error for

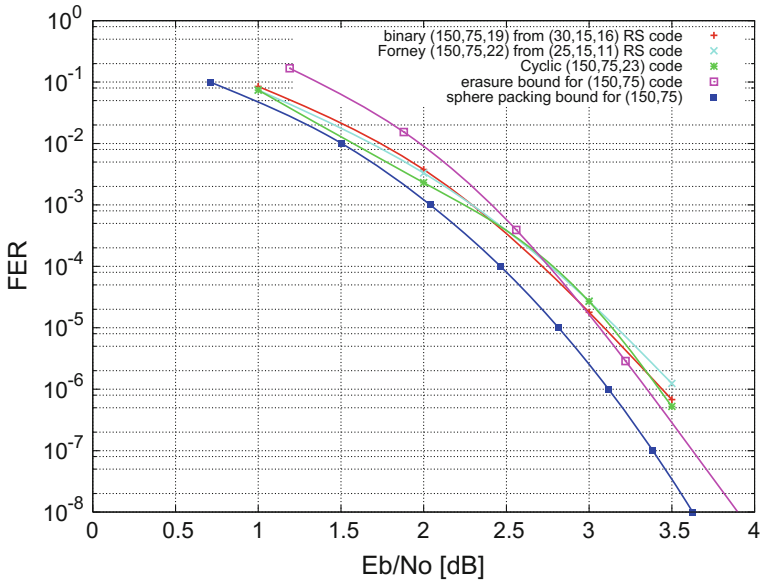

Fig. 7.3 Comparison of the (150, 75, 19) code derived from the RS(30, 15, 16) code, the concatenated (150, 75, 22) code and the known, best (150, 75, 23) code derived by shortening the (151, 76, 23) cyclic code

the three codes after 100 decoder error events. The results are shown in Table 7.2 at $\frac{E_b}{N_0} = 3$ dB. From Table 7.2 it can be seen that the concatenated code (150, 75, 22) has more error events with Hamming distances in the range 22–32, but the (150, 75, 23) known, best code has more error events for Hamming distances up to 36 compared to the (150, 75, 19) RS derived code, and this is the best code at $\frac{E_b}{N_0} = 3$ dB.

The distribution of error events is illustrated by the cumulative distribution of error events plotted in Fig. 7.4 as a function of Hamming distance. The weakness of the (150, 75, 22) code at $\frac{E_b}{N_0} = 3$ dB is apparent.

At higher values of $\frac{E_b}{N_0}$, the higher d_{min} of the (150, 75, 23) known, best code causes it to have the best performance as can be seen from Fig. 7.3.

7.4 Summary

This chapter studied further the Reed–Solomon codes which are ideal symbol-based codes because they are Maximum Distance Separable (MDS) codes. These codes are not binary codes but were considered for use as binary codes in this chapter. The performance of Reed–Solomon codes when used on a binary channel was explored and compared to codes which are designed for binary transmission. The construction of the parity-check matrices of RS codes for use as binary codes was described

Table 7.2 Hamming distances and multiplicities of 100 error events for each of the (150, 75) codes at $\frac{E_b}{N_0} = 3$ dB

Hamming distance	(150, 75, 19) Code number	(150, 75, 22) Code number	(150, 75, 23) Code number
22	0	4	0
24	0	4	0
25	1	0	0
26	1	9	0
27	3	0	7
28	5	7	6
29	4	0	0
30	8	22	0
31	7	0	15
32	7	19	20
33	14	0	0
34	8	14	0
35	5	0	19
36	8	13	18
37	3	0	0
38	8	5	0
39	7	0	9
40	6	1	2
41	2	0	0
42	1	2	0
43	1	0	1
44	1	0	1
47	0	0	1
48	0	0	1

in detail for specific code examples. The performance results of three differently constructed (150, 75) codes simulated for the binary AWGN channel, using a near maximum likelihood decoder, were presented. Surprisingly the best performing code at 10^{-4} error rate is not the best, known (150, 75, 23) code. Error event analysis was presented which showed that this was due to the higher multiplicities of weight 32–36 codeword errors. However, beyond 10^{-6} error rates the best, known (150, 75, 23) code was shown to be the best performing code.

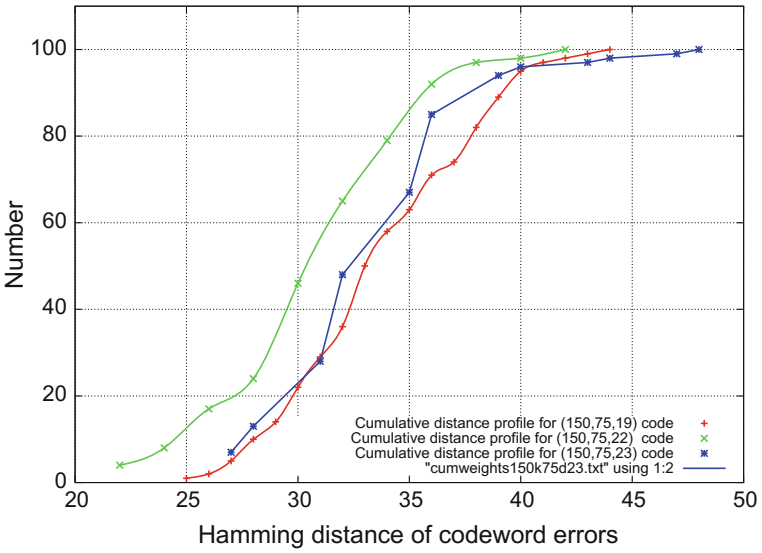

Fig. 7.4 Cumulative distribution of Hamming distance error events for the (150, 75, 19) code derived from the RS(30, 15, 16) code, the RS binary parity-check concatenated (150, 75, 22) code and the known, best (150, 75, 23) code derived by shortening the (151, 76, 23) cyclic code

References

1. El-Khamy, M., McEliece, R.J.: Iterative algebraic soft decision decoding of Reed–Solomon codes. Int. Symp. Inf. Theory Appl. **2004**, 1456–1461 (2004)
2. Grassl, M.: Code Tables: bounds on the parameters of various types of codes. http://www.codetables.de (2007)
3. Guruswami, V., Sudan, M.: Improved decoding of Reed–Solomon and algebraic-geometry codes. IEEE Trans. Inf. Theory **45**(6), 1757–1767 (1999)
4. Kasahara, M., Sugiyama, Y., Hirasawa, S., Namekawa, T.: New classes of binary codes constructed on the basis of concatenated codes and product codes. IEEE Trans. Inf. Theory IT-22 **4**, 462–468 (1976)
5. Koetter, R., Vardy, A.: Algebraic soft-decision decoding of Reed–Solomon codes. IEEE Trans. Inf. Theory **49**(11), 2809–2825 (2003)
6. MacWilliams, F.J., Sloane N.J.A.: The Theory of Error-Correcting Codes. North-Holland (1977)
7. McEliece, R.J.: The Guruswami–Sudan decoding algorithm for Reed–Solomon codes. JPL TDA Prog. Rep. **42**(153), 1–60 (2003)
8. Proakis, J.: Digital Communications, 4th edn. McGraw-Hill, New York (2001)
9. Reed, I., Solomon, G.: Polynomial codes over certain finite fields. J. Soc. Ind. Appl. Math. **8**, 300–304 (1960)
10. Sudan, M.: Decoding of Reed–Solomon codes beyond the error-correction bound. J. Complex. **13**, 180–193 (1997)